UNDERGRADUATE COMPUTER SECURITY EDUCATION

A Report on our Experience and Learning

Shiva Azadegan, Michael O'Leary, Alexander Wijesinha, and Marius Zimand

Towson University

Abstract: We describe our experiences from the first three years we have offered our track in computer security for our computer science major. We present the details of the track, including descriptions of the courses we have offered. We discuss the lessons we have learned offering the track, as well as the challenges that remain

Key words: Computer Security Education

1. INTRODUCTION

In Fall 2002 we offered our first course in our new Computer Security Track in the Computer Science major here at Towson University, and last year we graduated our second class of students. In this paper, we would like to describe our track and how it has developed as the program has grown.

Our track in computer security is our traditional Computer Science major where our students choose specific courses in computer security for their upper level electives. The computer security portion of the track is centered on the following seven courses:

- Computer Ethics,
- Introduction to Information Security,
- Introduction to Cryptography,
- Network Security,
- Application Software Security,
- Operating Systems Security, and
- Case Studies in Computer Security.

By design, our track focuses on the technical, practical, and applied areas of computer security. This design decision was made in part because this is a track within our regular computer science major; our students still take the core computer science courses like Computer Architecture, Operating Systems, and Database Management. The track courses are actually built upon the core courses. Figure 1 depicts the prerequisite tree for these courses. We also have a Master's degree with a concentration in computer security, but we will not discuss that concentration in this paper.

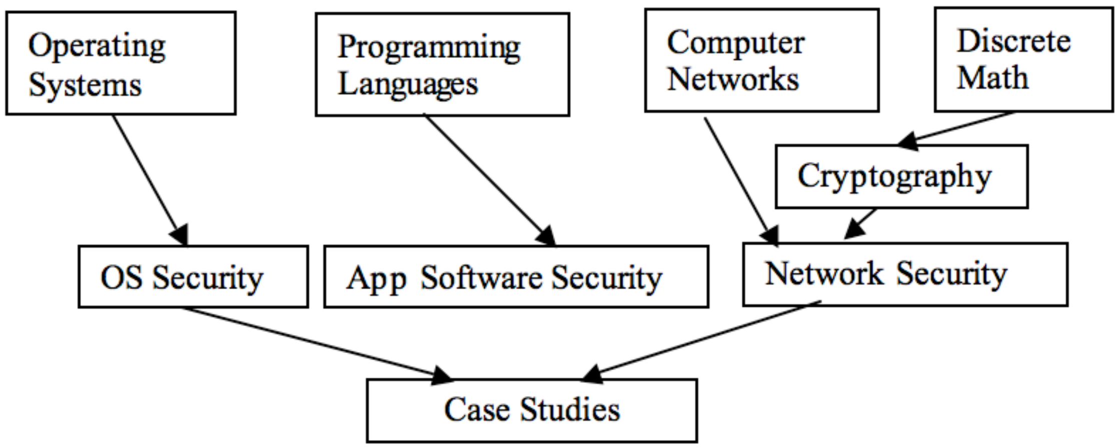

Figure 1. Computer Security Track Prerequisite Tree

Our initial approach to the design of our track can be found in [3,4]. Since then, a number of other approaches to integrating security into the curriculum have been presented. Perrone, Aburdene and Meng [15] present an informal survey of tracks and courses in computer security, while Tikekar [22] describes a new undergraduate track in computer security and information assurance at Southern Oregon University; see also [21]. Similarly, Dornseif *et. al*. [8,9] describe a data security program at a German university. A case study of the general process of information security curriculum development is presented by Bogolea and Wijekumar in [7].

As we enter the fourth year of existence for our computer security track here at Towson University, we have found that enrollment in our track has been healthy though small for a school of this size and has increased each year. Enrollment in our Cryptography course has increased from 15 in the track's first year, to 17 in the second to 21 last year. Enrollment in this course is an important barometer of the number of students who are entering the security track, as it is a prerequisite for the Network Security course, which in turn is a prerequisite for the Case Studies course; thus we recommend students in the track take this course early in their junior year. At the other end of the track is the Case Studies course, which is our capstone experience for the track, and taken in the spring semester of the senior year. Enrollment in this course was 5 in 2004, and nearly doubled to 9 in 2005; we expect that enrollment in 2006 will be even larger. Graduates from our track have been in great demand, and our former students have had little difficulty finding industry jobs quite quickly. Employers have been so pleased with our graduates, that they have contacted the faculty here directly, asking us to help them recruit more of our graduates.

2. FEATURES CONTRIBUTED TO THE SUCCESS OF THE TRACK

Based on our course evaluations and feedback from some of the employers of our students, the strong hands-on component of the security track has been the most valuable learning experience for our students. The courses in the track are built upon the core courses in the computer science program and they all have upper-level computer science courses as their prerequisite. Thus, students do have the theoretical knowledge, the maturity and good programming skills that are necessary to do elaborate and interesting applied projects. Though, developing such projects is very time consuming for the instructors, we believe that is imperative for any computer security training.

We also made the decision that the track to be focused and technical for the CS majors rather than broad and open to all; thus the students after graduating are skilled, confident and knowledgeable enough to be hired as information security officers.

All of the courses in the computer security track use our dedicated computer security laboratory for lab exercises. This is a physically secured room with a local network that is isolated, both from the campus as well as from the Internet. Student access to the laboratory is allowed outside of class only to students registered in one of our security courses. There are a number of different approaches one can take to the design of an isolated security laboratories; we mention [11, 14, 17, 18, 24, 25, 26]. See also [10, 27] for distributed security laboratories.

The lab itself contains 28 high end workstations running VMWare, and students do their lab work on virtual machines. One of the prime advantages of this approach is that it lets us use the classroom laboratory for a range of courses. Each class maintains their own set of virtual machines; changing the class is as simple as changing the virtual machines that are running. These virtual machines give our students the ability to experiment with a wide range of operating systems; We have used Windows 2000, Windows XP, Red Hat Linux, SUSE Linux, CentOS, and FreeBSD. VMWare also allows users to run more than one virtual machine on a host at the same time; we have run as many as six virtual machines simultaneously on one host. Further, because VMWare has the ability to set up virtual networks for the virtual machines on a host, we can let students set up their own small networks without the need for additional hardware.

Our choice of a flexible laboratory based on virtual machines has some disadvantages however. For example, we have not been able construct complex network topologies for the live exercises in our case studies course that you will find in, for example, [14, 18]. We have also limited ourselves to Intel computers; we do not have virtual machines that simulate a Mac or network hardware like a router or switch. Despite these limitations, the security laboratory has been an essential component of the early success of our track.

3. THE EXPERIENCES & LEARNING

In this section we share our experiences and learning and discuss the changes that were consequently made to the track. As explained in reference [3,4], two of the courses in the track, Computer Ethics and Introduction to Information Systems Security, were existing courses. The former is a required course for all our students and the latter is an upper-level elective course. This section focuses on the remaining five courses in the track that were developed and taught by the authors.

3.1 Cryptography

The Cryptography course is, roughly speaking, structured into three modules: (1) symmetric cryptography, (2) public-key cryptography, and (3) protocols and other applications. The course has been offered four times in the Fall semesters from 2002 to 2005 and the enrollment has grown from 12 to 21 students. The required textbook is *Introduction to Cryptography with Coding Theory* by Wade Trappe and Lawrence C. Washington [23]. In what follows we focus on several teaching aspects that we think require special attention.

Within the first module, some classical cryptosystems are covered such as: shift, affine, substitution, Vigenere, Hill, one-time pad ciphers (the module also discusses the most commonly used modern cryptosystems, DES, with its variants, and AES). We found it appropriate to use a relatively slow teaching pace in the exposition of classical cryptosystems (2.5 weeks). The simplicity of these systems allows the student to obtain a good understanding of the main security issues that a cryptosystem has to address, of the capabilities of different types of attacks, and of some basic principles that stand behind the design of any cryptosystem. This section of the course is also used to gently introduce some mathematical techniques, such as modular arithmetic and some notions from probability theory. Even though elementary, many students are not familiar with such mathematical tools, and they have the chance to see concrete applications of what otherwise may appear to be just mathematical abstractions.

We consider important that students understand and see the necessity of using rigorous mathematical formalism for the definition of security and secrecy as opposed to an ad-hoc and intuitive approach that does not provide any guarantees and is flatly hazardous in the context of cryptography. This is not easy to teach, however the one-time pad cipher allows for the exposition of *Shannon secrecy* and from here the instructor can make the transition to *statistical security*, and then to *computational security*. These notions, presented in the simple form of security against ciphertext-only attack, can be justified and illustrated with the classical cryptosystems that the students have learned.

We have attempted to alleviate the feeling that the course is math-heavy. For this reason, there are no long compact periods of time completely dedicated to mathematics. The mathematical notions are always introduced in the context of their utilization in cryptography systems and protocols. For example finite fields are introduced just after AES, the necessary elements of number theory are presented in conjunction with RSA, El-Gamal, Diffie-Hellman protocol, and so on.

The last module of the course on protocols and applications (such as bit commitment, zero-knowledge proofs, secret sharing, digital cash, electronic voting, coin flipping by telephone, etc.) should be very attractive for the students, since they apparently imply "impossible" things rendered feasible through clever combinations of rather elementary mathematical notions that have been presented earlier in the course. Unfortunately, this module is rushed at the end of the course when students focus more on their project and on the preparation for the final exam. Consequently we never had the time to cover but one or two applications from the above list and we felt that the students did not digest all the subtleties of these applications. We think that it is possible to teach parts of the second module more aggressively and to insert some of the applications at different points in the course.

The assignments, generally speaking, fall into three large categories: (a) math exercises needed to fix the notions and to develop mathematical skills, (b) concrete attacks on "small" implementations of the crypto systems and crypto protocols covered in class, and (c) exercises in which students are asked to analyze variations of crypto systems and crypto protocols and to reveal the weaknesses of the proposed variations. The textbook is a good source for exercises in the categories (a) and (c).

The list of suggested projects takes into account that the course audience includes students that have little or zero programming experience. Thus the list includes: (a) several programming projects (such as the implementation of differential attack on a small version of DES, RSA, different signature schemes, etc.), (b) projects that involve reading recent research articles and writing a survey paper, and (c) projects that ask the student to design protocols for some cryptography functionality using concrete "real-world" objects such as boxes with different kind of locks, pebbles, etc. Such protocols are used in cryptography as physical metaphors for digital implementations of the functionality and serve as a first intuitive step in the design of the protocol. One example of such a project is the following. Alice has a number n_a and Bob has a number n_b, both in the set of integers ranging from 1 to 100. They want to know which one has a larger number but they do not want to reveal any other information (or as little information as possible). They can use boxes and pebbles in their protocol. The pebbles are identical (they produce the same sound when the boxes are shaken). Your task is to design such a protocol and to analyze it. In your analysis, consider the number of pebbles, the number of boxes, and how much information is leaked. Ideally, if Alice learns that, say, n_b is larger than n_a, then, from her point of view, all the numbers larger than n_a should be equally likely to be n_b. For example consider that $n_a = 20$ and that at the end of the protocol, Alice knows that n_b is larger than *20*, that is n_b is one of the numbers *21, 22, …, 100*. Then ideally $Prob(n_b = 21) = Prob(n_b = 22) = \ldots = Prob(n_b = 100)$. Solutions in which the probabilities are not equal but are close are acceptable (let's say the difference of any two probabilities in absolute value is at most *0.01*).

3.2 Network Security

This course covers the principles of network security, focusing on specific application areas such as authentication (Kerberos and X.509 certificates), email security (PGP and S/MIME), IP security (IPSec), transport layer security (TLS/SSL), and firewalls. The course begins with a general overview of common attacks and security mechanisms and services for attack prevention and detection, followed by a two-week introduction to basic cryptography. The required textbook is Network Security Essentials by William Stallings [19]. Students working in groups are responsible for a paper/presentation and completion of 3-4 assignments. The group size depends on enrollments; there are typically 3-4 members per group. In determining the course grade, the four components midterm, final, paper/presentation and assignments have equal weight. The course has been offered 6 times from Spring 2003 through Fall 2005, once as an independent study, and has seen a maximum enrollment of 10 students.

The material covered in the course is fairly detailed and the objective is to provide students with an understanding of the various methodologies and security protocols employed in network security. Exam questions have included a combination of short answer, multiple choice and problems.

The instructor assigns a topic for the paper to each group. Topics have included wireless LAN (802.11) security, cellular network security, denial of service attacks, and IP, ICMP and TCP vulnerabilities. The paper is written in a formal conference style requiring 4-5 pages in two-column format. While the paper is not expected to be overly technical, it is to be written at about the same level of detail as topics covered in the course text. The paper also provides an opportunity to assign topics in network security that are of current interest and address new developments in the field.

The assignments constitute the most interesting part of the course. For their first assignment, students use a socket program to implement the Diffie-Hellman exchange and transfer a message over a network encrypted using AES. They could write their own code, or use prewritten free software and tools available on the Web. Other assignments include using GPG and its trust model to transmit secure email, and using Snort to log packets, write rules and trigger alerts based on network protocols and message characteristics.

When the course was originally proposed, SNMP security was included as a topic. However, since students would need some background in network management and SNMP, it was decided to drop this topic due to a lack of time. The system security aspects of the course are limited to a discussion of firewalls from a network viewpoint. The reason is that topics such as viruses and worms, intrusion detection and honeynets are covered in other courses in the track. Both SNMP security and system security topics could be assigned as topics for the paper.

Network security has often been taught as a combined undergraduate/graduate course. This impacted the undergraduate course in that the material had to be presented at a somewhat higher level with a little more emphasis on the technical details. The differences in the level and background of the graduate and undergraduate students also created some difficulties. While the topics covered during lectures were the same for both courses, the graduate students were assigned additional papers for independent study and a semester project that was significantly more complex and required a substantial amount of programming. Presently, the undergraduate course is not being combined with the graduate course.

An issue that needs further discussion is that of prerequisites for the network security course. At first, a course in cryptography and a course in computer networks were prerequisites. However, since an overview of cryptography is presented at the beginning of the network security course, it was felt that the cryptography prerequisite may be omitted. Currently, only a course in computer networks is required.

Overall, the course has been quite successful judging from end-of-semester student evaluations. Assignments that address IP/TCP vulnerabilities, IPSec, SSL/TLS and wireless security would be beneficial. Unfortunately, we have not yet found assignments based on these topics that are non-trivial and possible to complete in approximately two weeks. We would also like to include more assignments that involve programming and require students to understand code that implement network security protocols.

3.3 Application Software Security

The Application Software Security course introduces students to the security concepts in developing software applications. This course discusses design principles for secure software development, and some of the security issues in current programming and scripting languages, database systems and web servers. This course is the only one that has been taught by more that one instructor. Three instructors with approaches varying from small number of hands-on projects and strong current literature research projects, to equal time allocated to lab and lecture, to completely lab-based, starting with the overview of Assembly language and run-time environment, have taught the course. The completely lab-based approach is being experimented during this semester (Fall 05), though we don't have the final data, from the preliminary feedback of the students, they all enjoy the course greatly. We do firmly believe that like other courses in this track, a strong lab component is necessary. Some of the topics and projects covered in this class include:

– Buffer overflow – Students are presented with an in-depth discussion of stack and heap overflow problems and how to exploits are generated. The reading assignments include papers addressing the overflow problem, integer and format string overflow problems, methods of defense against buffer overflow and secure programming techniques to prevent buffer overflow. In the lab, students work with relatively simple programs with buffer overflow vulnerability and asked to write the exploits. Though our computer science majors do take a course in Assembly language, we found that our undergraduate students don't have the needed skills to actually handle more challenging projects in this area.

– Threat modeling – There is a good coverage of this topic in "writing secure code" [12]. Microsoft has developed a tool [2] that is freely available and can be used to make the projects on this topic more interesting and challenging. Threat Modeling [20] provides a good reference for this project.

– Authentication and authorization – we provide students with a broad overview of authentication, authorization and access control techniques. Java security model is presented here and students' projects deal with using JAAS. Also we discussed Kerberos in details and a simple project dealing with configuring a Kerberos server in our security lab and using it to authenticate users for their projects.

– Cross-site scripting and SQL injection – Students enjoyed the discussion of both topics. Students did team projects demonstrating these problems.

3.4 Operating Systems Security

The operating systems security course introduces students to operating systems security issues and how to secure a system, and it has Operating Systems as a prerequisite. When the course was originally designed, we allocated equal time to Unix and Windows systems. However, based on our experience, we spend more time discussing Unix system than Windows. First, we realized that our students are less familiar with the Unix environment than Windows. Thus, we had to spend few weeks at the beginning to discuss the Unix operating system. Second, there are much better textbooks, articles and resources available addressing Unix security vulnerabilities than those for Windows. We tried to cover the same topics for both systems, to reinforce the concepts and allow students to make a better comparison of the features.

The coursework for this class consists of literature review, projects and exams. The students were assigned a paper each week to read and had to write a one-page review. Examples include "Security Report: Windows vs. Linux" [16], "Root Kits – An Operating Systems Viewpoint" [13], and "Leave no Trace, playing Hide and Seek Unix Style" [4]. Students very much enjoyed reading and reporting on these papers.

For the exercises and projects in this class, students had root or administrator access to their virtual machine. There are 7-8 projects small team projects in this class in addition to the final term project that depends on its scope, and can be either an individual or a team project. The projects dealing with Unix environment include writing a password cracker program; creating and managing user accounts; installing, configuring and using Tripwire [2]; configuring a simple secure ftp site and creating "jails"; and hardening Linux project using Bastille. We also used the NSA Security Configuration Guides for Windows 2000 [1] and found them very helpful and complete.

3.5 Case Studies in Computer Security

The Case Studies in Computer Security course serves as the capstone experience for the security track. Only offered in the spring semester, it has Operating Systems Security and Network Security as prerequisites. This course has been offered twice, in Spring 2004 and Spring 2005 to five and nine undergraduate students. It is a hands-on course that emphasizes defense, detection, and administration, and is arranged around a sequence of five or six competitive team-based laboratory exercises in the security laboratory. In a typical exercise, four different teams of students design and construct their own network of machines. They can choose from a range of operating systems, but their resulting network is required to provide a suite of remote services like Web, SSH, or FTP to their competing teams in the laboratory. Each team is then provided with authentication credentials to one or more of the services offered by some of the other teams, with the conditions that

- No team has root equivalent credentials on any machine from another team,
- No team has all of the non-root credentials for any other team, and
- No team knows which other teams have credentials for the services that they are to provide.
–

During the live portion of the exercise, each team must verify that the services provided by opposing teams for which they have authentication credentials are correctly functioning. They then try to illicitly gain access to all of the remaining services and the remote host itself, if possible. Once the live portion of the exercise has completed, students review their logs to try to determine who accessed their network, and whether they did so legitimately or illegitimately.

The purpose of the course is to give our graduates experience configuring and operating a network of machines in an unknown environment. Although attacks, and attack tools are described, the majority of the class time is spent learning defensive skills. Topics covered in the class include:

– Account and password management. PAM, password cracking.

– Logging and Auditing. Setting up a log server.

– Simple reconnaissance techniques; ping, nmap.

– Packet sniffers; Ethereal.

– Intrusion detection systems; Snort.

– Configuring common services: IIS, Apache, OpenSSH, WU-FTP.

– Advanced reconnaissance: Null connections and NetBIOS enumeration, SNMP walking.

– Backdoors: netcat, vnc.

– Firewalls. Iptables.

– Security analysis tools: Nessus, Microsoft baseline security analyzer.

– Security configuration tools: Bastille, Microsoft IIS lockdown tool.

For attacks, we present attack tools like Metasploit and the sniffer/password cracker Cain & Abel. The decision to give less weight to attacks and more weight to defense is not motivated by philosophical reasons, but rather by practical considerations. There is simply not enough time in the course to cover everything, and the purpose of the course is to teach potential security officers and system administrators- not penetration testers. For example, we have discovered as the class has been taught that it takes students a great deal of time- three or four weeks- to learn how to write good firewall rulesets, especially when they are to govern a realistic network with some complexity.

As an example of our live exercises, let us briefly describe the course's final exercise. In it, each of the four teams is told that they are the IT department for a hypothetical company. The company has offices in three different physical locations and they need to be able to work collaboratively on projects. The company also needs simple web presence, and the ability to deliver documents to the public. Finally the company has a corporate partner that should have access to additional information not available to the public, as well as the ability to work collaboratively on projects. The role of the partner company will be played by one of the other three teams. With these general guidelines, student teams are free to construct their own network and choose what services they will provide to meet these business requirements. Students begin by designing their network; in addition to setting up workstations to represent each office, they set up a collection of servers to provide the company's public presence. The students also set up and configure one or more logging servers, firewalls, and intrusion detection systems. All together, the preparation takes around two weeks. In the live portion of the exercise students access the services provided by both their partners and their competitors. Once the live portion of the exercise concludes, students write a report where they describe what they did to their competitors, and more importantly what their competitors did to them. Their project grade is primarily based on how well they set up their network, and how well they detect the attacks that other teams send their way.

4. ADMINISTRATIVE CHALLENGES

Like all new tracks, our security track has had its share of administrative challenges. Scheduling courses needed for completion of the track has presented some difficulties. Currently, we are offering courses according to the following plan:

– Fall: Cryptography, Network Security, Operating System Security

– Spring: Application Software Security, Case Studies

In the past, however, Operating Systems Security, Application Software Security and Network Security have been offered in both Fall and Spring. This has been due to several reasons including newness of the track, prerequisites, student demand, expediting graduation, and faculty availability.

Another factor that has impacted scheduling in the past is the offering of combined undergraduate and graduate classes. During the first few semesters the track was offered, some classes had few students. This was to be expected until such time that new students in the track would become juniors or seniors and start taking the security courses. In fact, at the beginning, many students taking these classes were not in the security track. Some were not even computer science majors. This together with the combined undergraduate and graduate enrollments ensured that class sizes were administratively acceptable. Now that the track has been available for a few years and student awareness of the benefits of completing the track has increased, we expect reasonably steady enrollments in all courses.

At present, there are only five faculty members teaching the security courses. All are tenured or tenure-track; one has a joint appointment in Mathematics and Computer Science; the others are in Computer Science. With the exception of Application Software Security, the same faculty member has always taught every offering of a given course. So far, this has worked out well, since the courses have matched the interests and expertise of the faculty. It also allowed courses to be fine-tuned and improved with each offering, and provided continuity. However, this approach presents difficulties in that faculty availability and track schedules may not always match. Although the department currently has about 24 tenured/tenure-track faculty members, most have interests in other areas of Computer Science and Information Systems, which means there is little flexibility in assigning instructors to security courses.

5. FUTURE PLANS

As we enter the fourth year of the track, we have begun a complete review of the track and its courses. Currently, we are considering a number of modifications and improvements to the track. They include:

– Changing the prerequisite tree for the track. In particular, we are considering removing the prerequisite course in Cryptography from the Network Security course; this will reduce the size of the (lengthy) chain of courses that our students need to graduate. As it stands, Cryptography is a prerequisite for Network Security, which is a prerequisite for Case Studies in Computer Security; this long chain of 300 and 400 level courses has been a difficulty for students.

– We are assessing the contribution of the introductory course, Introduction to Information Systems Security, to the whole track. This course provides a broad overview of technical and human components of information systems security. Therefore, there is a considerable overlap and redundancy between this course and the remaining courses in the track We are considering the possibility of reusing the credit hours to design a new course.

– We may want to designate one of the sections of computer ethics course for the security track students to specifically address some of the security policies and regulations. Right now, the Computer Ethics course is a general course for all of our majors, and we may want to use this course to emphasize the ethical issues that arise in computer security.

– We are trying to forge closer connections between the different courses. Some topics occur naturally in different courses; for example firewalls and intrusion detection systems are described in both the Network Security course and the Case Studies course. We would are trying to ensure that the topic is covered in a complementary fashion in the two courses.

Another issue that has arisen is the rapid pace of change in the field. Given the evolving nature of computer security, to what extent do course assignments and projects need to reflect ongoing developments? For example, the weaknesses of 802.11 WEP are well known. Should students be given a hands-on assignment using WEP and the interim WPA? What elements of the proposed 802.11i security standard need to be included? Should the topic of wireless security be assigned as a paper topic in the network security class, or should it be included as one of the topics covered in the lectures?

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under grant No. DUE-0113783

REFERENCES

1. NSA Security Configuration Guides for Windows 2000. Internet http://www.nsa.gov/snac/index.cfm
2. Threat Modeling Tool. Internet http://www.microsoft.com/downloads/details.aspx?familyid=62830f95-0e61-4f87-88a6-e7c663444ac1&displaylang=en. Accessed September, 2005.
3. Tripwire. Internet http://sourceforge.net/projects/tripwire/ Accessed September 2005.
4. Accident, P. Playing Hide and Seek, Unix style. Phrack Magazine, Issue 43, File 14. Internet: http://www.phrack.org/phrack/43/P43-14 Accessed September 2005.
5. Azadegan, S., Lavine, M., O'Leary, M., Wijesinha, A., and Zimand, M. 2003. A dedicated undergraduate track in computer security education. In Security Education and Critical infrastructures, C. Irvine and H. Armstrong, Eds. Kluwer Academic Publishers, Norwell, MA, 319-331.
6. Azadegan, S., Lavine, M., O'Leary, M., Wijesinha, A., and Zimand, M. 2003. An undergraduate track in computer security. In Proceedings of the 8th Annual Conference on innovation and Technology in Computer Science Education (Thessaloniki, Greece, June 30 - July 02, 2003). D. Finkel, Ed. ITiCSE '03. ACM Press, New York, NY, 207-210.
7. Bogolea, B. and Wijekumar, K. Information security curriculum creation: a case study. In InfoSecCD '04: Proceedings of the 1st annual conference on Information security curriculum development, pages 59--65, New York, NY, USA, 2004. ACM Press.
8. Dornseif, M., Freiling, F. (geb. Gärtner), Holz, T. and Mink, M. An offensive approach to teaching information security: "Aachen summer school applied it security". Technical Report AIB-2005-02, Aachen, 2005.
9. Dornseif, M., Gaertner, F.C., Mink, M. and Pimenidis, L. Teaching data security at university degree level. Proceedings of the WISE04 (to appear). Internet: http://md.hudora.de/publications/.
10. Fulp, J. The bastion network project. In C. Irvine and M. Rose, editors, Avoiding Fear, Uncertainty and Doubt Through Effective Security Education, pages 65--71. Center for Information Systems Security Studies and Research, Naval Postgraduate School, 2004.

11. Hill, J.M.D., Curtis, J., Carver, A., Humphries, J.W., and Pooch, U.W. Using an isolated network laboratory to teach advanced networks and security. In SIGCSE '01: Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education, pages 36--40, New York, NY, USA, 2001. ACM Press.
12. Howard, M. and LeBlanc, D. Writing Secure Code, Microsoft Press, 2003.
13. Kühnhauser, W. E. 2004. Root Kits: an operating systems viewpoint. SIGOPS Oper. Syst. Rev. 38, 1 (Jan. 2004), 12-23.
14. Lathrop, S.D., Conti, G.J., and Ragsdale, D.J. 2003. Information warfare in the trenches. In Security Education and Critical infrastructures, C. Irvine and H. Armstrong, Eds. Kluwer Academic Publishers, Norwell, MA, 19-39.
15. Perrone, L. F., Aburdene, M., and Meng, X. Approaches to undergraduate instruction in computer security. Proceedings of the American Society for Engineering Education Annual Conference and Exhibition, ASEE 2005. Internet: http://www.ists.dartmouth.edu/library/116.pdf.
16. Petreley, N. Security Report: Windows vs Linux, The Register. Internet http://www.theregister.co.uk/security/security_report_windows_vs_linux/ Accessed September 2005.
17. Romney, G.W. and Stevenson, B.R. An isolated, multi-platform network sandbox for teaching it security system engineers. In CITC5 '04: Proceedings of the 5th conference on Information technology education, pages 19--23, New York, NY, USA, 2004. ACM Press.
18. Schafer, J., Ragsdale, D.J., Surdu, J.R., and Carver, C.A. The IWAR range: a laboratory for undergraduate information assurance education, Proceedings of the sixth annual CCSC northeastern conference on The journal of computing in small colleges, p.223-232, April 2001, Middlebury, Vermont, United States.
19. Stallings, W. Network Security Essentials: Applications and Standards. Prentice Hall, 2nd Edition, 2003.
20. Swiderski, F. and Snyder, W. Threat Modeling. Microsoft Press, 2004.
21. Tikekar, R. and Bacon, T. The challenges of designing lab exercises for a curriculum in computer security. J. Comput. Small Coll., **18**(5):175--183, 2003.
22. Tikekar., R.V. Teaching computer security to undergraduates. In C. Irvine and M. Rose, editors, Avoiding Fear, Uncertainty and Doubt Through Effective Security Education}, pages 5--10. Center for Information Systems Security Studies and Research, Naval Postgraduate School, 2004.
23. Trappe, W., and Washington L. Introduction to Cryptography with Coding Theory, Prentice Hall, 2006 (second edition).
24. Vigna, G. Teaching Hands-On Network Security: Testbeds and Live Exercises. Journal of Information Warfare, **3**(2):8--25, 2003.
25. Vigna, G. Teaching Network Security Through Live Exercises. In C. Irvine and H. Armstrong, editors, Proceedings of the Third Annual World Conference on Information Security Education (WISE 3), pages 3--18, Monterey, CA, June 2003. Kluwer Academic Publishers.
26. Wagner, P.J. and Wudi, J.M. Designing and implementing a cyberwar laboratory exercise for a computer security course. In Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education (Norfolk, Virginia, USA, March 03 - 07, 2004). SIGCSE '04. ACM Press, New York, NY, 402-406.
27. Yang, T. A., Yue, K., Liaw, M., Collins, G., Venkatraman, J. T., Achar, S., Sadasivam, K., and Chen, P. Design of a distributed computer security lab. J. Comput. Small Coll. 20, 1 (Oct. 2004), 332-346.